HW6_Q1

November 1, 2019

[15]:
```
from gurobipy import *
```

1 Q1

[16]:
```
f = open("shortest_path_data-1.txt", "r")
lines = f.readlines()[1:]
myModel = Model("Shortest_path")
cost = [[0 for i in range (8)] for j in range (8)]
myVars = [[0 for i in range (8)] for j in range (8)]
for i in lines:
    initial = i.split()
    origin = int (initial[0])
    destination = int (initial[1])
    cost [origin - 1][destination - 1] = float (initial[2])
print (cost)
```

```
[[0, 1.0, 2.0, 0, 0, 0, 0, 0], [0, 0, 1.0, 5.0, 2.0, 0, 0, 0], [0, 0, 0, 2.0,
1.0, 4.0, 0, 0], [0, 0, 0, 0, 3.0, 6.0, 8.0, 0], [0, 0, 0, 0, 0, 3.0, 7.0, 0],
[0, 0, 0, 0, 0, 0, 5.0, 2.0], [0, 0, 0, 0, 0, 0, 0, 6.0], [0, 0, 0, 0, 0, 0, 0,
0]]
```

[17]:
```
for i in range(8):
    for j in range(8):
        curVar = myModel.addVar(vtype = GRB.CONTINUOUS, name = "X" + str(i) +␣
 ↪str(j))
        myVars[i][j] = curVar
myModel.update()
```

[18]:
```
objExpr = LinExpr()
for i in range(8):
    for j in range(8):
        curVar = myVars[i][j]
        objExpr += cost[i][j] * curVar

myModel.setObjective(objExpr, GRB.MINIMIZE)
```

```
print(objExpr)
```

```
<gurobi.LinExpr: 0.0 X00 + X01 + 2.0 X02 + 0.0 X03 + 0.0 X04 + 0.0 X05 + 0.0 X06
+ 0.0 X07 + 0.0 X10 + 0.0 X11 + X12 + 5.0 X13 + 2.0 X14 + 0.0 X15 + 0.0 X16 +
0.0 X17 + 0.0 X20 + 0.0 X21 + 0.0 X22 + 2.0 X23 + X24 + 4.0 X25 + 0.0 X26 + 0.0
X27 + 0.0 X30 + 0.0 X31 + 0.0 X32 + 0.0 X33 + 3.0 X34 + 6.0 X35 + 8.0 X36 + 0.0
X37 + 0.0 X40 + 0.0 X41 + 0.0 X42 + 0.0 X43 + 0.0 X44 + 3.0 X45 + 7.0 X46 + 0.0
X47 + 0.0 X50 + 0.0 X51 + 0.0 X52 + 0.0 X53 + 0.0 X54 + 0.0 X55 + 5.0 X56 + 2.0
X57 + 0.0 X60 + 0.0 X61 + 0.0 X62 + 0.0 X63 + 0.0 X64 + 0.0 X65 + 0.0 X66 + 6.0
X67 + 0.0 X70 + 0.0 X71 + 0.0 X72 + 0.0 X73 + 0.0 X74 + 0.0 X75 + 0.0 X76 + 0.0
X77>
```

[19]:
```
constExpr = LinExpr()
for j in range(8):
    if cost[0][j] != 0:
        constExpr += 1 * myVars[0][j]
myModel.addConstr(lhs = constExpr, sense = GRB.EQUAL, rhs = 1)

constExpr = LinExpr()
for i in range(8):
    if cost[i][7] != 0:
        constExpr += 1 * myVars[i][7]
myModel.addConstr(lhs = constExpr, sense = GRB.EQUAL, rhs = 1)

for i in range(1,7):
    constExpr = LinExpr()
    for j in range(8):
        if cost[i][j] != 0:
            constExpr += 1 * myVars[i][j]
        if cost[j][i] != 0:
            constExpr -= 1 * myVars[j][i]
    myModel.addConstr (lhs = constExpr, sense = GRB.EQUAL, rhs = 0)

myModel.update
```

[19]:
```
<bound method Model.update of <gurobi.Model Continuous instance Shortest_path: 0
constrs, 64 vars, Parameter changes: LogFile=gurobi.log, CSIdleTimeout=1800>>
```

[20]:
```
myModel.write(filename = "Shortest_path.lp")
myModel.optimize()
print("Optimal Objective: \n" + str(myModel.ObjVal))
print("Optimal Solution:")
allVars = myModel.getVars()
for curVar in allVars:
    print(curVar.varName + " " + str(curVar.x))
```

```
Optimize a model with 8 rows, 64 columns and 32 nonzeros
Coefficient statistics:
```

```
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+00, 8e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
Presolve removed 2 rows and 52 columns
Presolve time: 0.01s
Presolved: 6 rows, 12 columns, 24 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    3.9920000e+00   1.503000e+00   0.000000e+00      0s
       3    8.0000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 3 iterations and 0.02 seconds
Optimal objective  8.000000000e+00
Optimal Objective:
8.0
Optimal Solution:
X00 0.0
X01 1.0
X02 0.0
X03 0.0
X04 0.0
X05 0.0
X06 0.0
X07 0.0
X10 0.0
X11 0.0
X12 1.0
X13 0.0
X14 0.0
X15 0.0
X16 0.0
X17 0.0
X20 0.0
X21 0.0
X22 0.0
X23 0.0
X24 1.0
X25 0.0
X26 0.0
X27 0.0
X30 0.0
X31 0.0
X32 0.0
X33 0.0
X34 0.0
X35 0.0
X36 0.0
```

```
X37 0.0
X40 0.0
X41 0.0
X42 0.0
X43 0.0
X44 0.0
X45 1.0
X46 0.0
X47 0.0
X50 0.0
X51 0.0
X52 0.0
X53 0.0
X54 0.0
X55 0.0
X56 0.0
X57 1.0
X60 0.0
X61 0.0
X62 0.0
X63 0.0
X64 0.0
X65 0.0
X66 0.0
X67 0.0
X70 0.0
X71 0.0
X72 0.0
X73 0.0
X74 0.0
X75 0.0
X76 0.0
X77 0.0
```

[ ]:

```
\ Model Shortest_path
\ LP format - for model browsing. Use MPS format to capture full model detail.
Minimize
  0 X00 + X01 + 2 X02 + 0 X03 + 0 X04 + 0 X05 + 0 X06 + 0 X07 + 0 X10
   + 0 X11 + X12 + 5 X13 + 2 X14 + 0 X15 + 0 X16 + 0 X17 + 0 X20 + 0 X21
   + 0 X22 + 2 X23 + X24 + 4 X25 + 0 X26 + 0 X27 + 0 X30 + 0 X31 + 0 X32
   + 0 X33 + 3 X34 + 6 X35 + 8 X36 + 0 X37 + 0 X40 + 0 X41 + 0 X42 + 0 X43
   + 0 X44 + 3 X45 + 7 X46 + 0 X47 + 0 X50 + 0 X51 + 0 X52 + 0 X53 + 0 X54
   + 0 X55 + 5 X56 + 2 X57 + 0 X60 + 0 X61 + 0 X62 + 0 X63 + 0 X64 + 0 X65
   + 0 X66 + 6 X67 + 0 X70 + 0 X71 + 0 X72 + 0 X73 + 0 X74 + 0 X75 + 0 X76
   + 0 X77
Subject To
 R0: X01 + X02 = 1
 R1: X57 + X67 = 1
 R2: - X01 + X12 + X13 + X14 = 0
 R3: - X02 - X12 + X23 + X24 + X25 = 0
 R4: - X13 - X23 + X34 + X35 + X36 = 0
 R5: - X14 - X24 - X34 + X45 + X46 = 0
 R6: - X25 - X35 - X45 + X56 + X57 = 0
 R7: - X36 - X46 - X56 + X67 = 0
Bounds
End
```

Q2(a)

u_i : unit cost from central depot to warehouse

X_{ij} : number of units from warehouse i to retailer j

c_{ij} : unit cost from warehouse i to retailer j

D_j : demand of retailer j

$$\min \sum_{i=1}^{10} u_i * \sum_{i=1}^{10}\sum_{j=1}^{15} X_{ij} + \sum_{i=1}^{10}\sum_{j=1}^{15} c_{ij}X_{ij}$$

$$st. \sum_{i=1}^{10} X_{ij} = D_j, \quad \forall j = 1,2,3,\dots,15$$

$$\sum_{i=1}^{10}\sum_{j=1}^{15} X_{ij} = 615$$

$$X_{ij} \leq 10, X_{ij} \geq 0$$

HW6_Q2

November 1, 2019

[72]:
```
from gurobipy import *
```

1 Q2(b)

[73]:
```
cost = []
with open('supply_chain_data.txt', 'r') as f:
    for line in f:
        cost.append(list(map(float,line.split())))
```

[74]:
```
central = cost.pop(0)
demand = cost.pop(-1)
total = sum(demand)
```

[75]:
```
myModel = Model ("Supply_chain")
Var = []
myVars = [[0 for i in range(len(demand))] for j in range(len(central))]

for i in range(len(central)):
    curVar = myModel.addVar(vtype = GRB.CONTINUOUS, name = "X" + str(0) + str(i␣
 ↪+ 1))
    Var.append(curVar)

for i in range(len(central)):
    for j in range(len(demand)):
        curVar=myModel.addVar(vtype=GRB.CONTINUOUS, name = "X" + str(i + 1) +␣
 ↪str(j + 1), ub = 10)
        myVars[i][j]=curVar

myModel.update()
```

[76]:
```
objExpr=LinExpr()
for i in range(len(central)):
    curVar=Var[i]
    objExpr += central[i]*curVar

for i in range(len(central)):
    for j in range(len(demand)):
```

```
        curVar=myVars[i][j]
        objExpr += cost[i][j]*curVar


myModel.setObjective(objExpr, GRB.MINIMIZE)
myModel.update()
print(objExpr)
```

```
<gurobi.LinExpr: 0.52 X01 + 0.55 X02 + 0.4 X03 + 0.39 X04 + 0.2 X05 + 0.13 X06 +
0.32 X07 + 0.14 X08 + 0.12 X09 + 0.69 X010 + 0.29 X11 + 0.04 X12 + 0.62 X13 +
0.59 X14 + 0.71 X15 + 0.96 X16 + 0.77 X17 + 0.96 X18 + 0.73 X19 + 0.06 X110 +
0.91 X111 + 0.44 X112 + 0.82 X113 + 0.55 X114 + 0.51 X115 + 0.19 X21 + 0.31 X22
+ 0.99 X23 + 0.53 X24 + 0.74 X25 + 0.12 X26 + 0.08 X27 + 0.54 X28 + 0.2 X29 +
0.83 X210 + 0.62 X211 + 0.1 X212 + 0.82 X213 + 0.64 X214 + 0.42 X215 + 0.77 X31
+ 0.91 X32 + X33 + 0.29 X34 + 0.29 X35 + 0.38 X36 + 0.95 X37 + 0.25 X38 + 0.25
X39 + 0.5 X310 + 0.19 X311 + 0.27 X312 + 0.36 X313 + 0.94 X314 + 0.02 X315 +
0.03 X41 + 0.64 X42 + 0.48 X43 + 0.23 X44 + 0.68 X45 + 0.76 X46 + 0.6 X47 + X48
+ 0.92 X49 + 0.1 X410 + 0.58 X411 + 0.21 X412 + 0.13 X413 + 0.98 X414 + 0.23
X415 + 0.51 X51 + 0.56 X52 + 0.92 X53 + 0.04 X54 + 0.63 X55 + 0.83 X56 + 0.49
X57 + 0.02 X58 + 0.85 X59 + 0.04 X510 + 0.69 X511 + 0.87 X512 + 0.33 X513 + 0.56
X514 + 0.82 X515 + 0.15 X61 + 0.47 X62 + 0.14 X63 + 0.85 X64 + 0.82 X65 + 0.78
X66 + 0.29 X67 + 0.27 X68 + 0.49 X69 + 0.82 X610 + 0.05 X611 + 0.13 X612 + 0.5
X613 + 0.72 X614 + 0.32 X615 + 0.79 X71 + 0.56 X72 + 0.61 X73 + 0.03 X74 + 0.53
X75 + 0.13 X76 + 0.23 X77 + 0.9 X78 + 0.98 X79 + 0.7 X710 + 0.03 X711 + 0.89
X712 + 0.31 X713 + 0.25 X714 + 0.42 X715 + 0.2 X81 + 0.7 X82 + 0.38 X83 + 0.54
X84 + 0.35 X85 + 0.92 X86 + 0.47 X87 + 0.77 X88 + 0.83 X89 + 0.53 X810 + 0.62
X811 + 0.19 X812 + 0.02 X813 + 0.41 X814 + 0.12 X815 + 0.58 X91 + 0.75 X92 +
0.57 X93 + 0.14 X94 + 0.1 X95 + 0.13 X96 + 0.01 X97 + 0.66 X98 + 0.84 X99 + 0.75
X910 + 0.39 X911 + 0.6 X912 + 0.55 X913 + 0.99 X914 + 0.04 X915 + 0.57 X101 +
0.76 X102 + 0.04 X103 + 0.39 X104 + 0.18 X105 + 0.25 X106 + 0.52 X107 + 0.31
X108 + 0.17 X109 + 0.04 X1010 + 0.81 X1011 + 0.86 X1012 + 0.58 X1013 + 0.99
X1014 + 0.65 X1015>
```

[77]:
```
for cons in range(len(demand)):
    constExpr = LinExpr()
    for j in range(len(myVars)):
        constExpr += myVars[j][cons]
    myModel.addConstr(lhs = constExpr, sense = GRB.EQUAL, rhs = demand[cons])

for cons in range(len(myVars)):
    constExpr = LinExpr()
    for j in range(len(demand)):
        constExpr += myVars[cons][j]
    constExpr -= Var[cons]
    myModel.addConstr(lhs = constExpr, sense = GRB.EQUAL, rhs = 0)

constExpr = LinExpr()
```

```
for i in range(len(Var)):
    constExpr += Var[i]

myModel.addConstr(lhs = constExpr, sense = GRB.LESS_EQUAL, rhs = 615)

myModel.update()
```

[78]:
```
myModel.write(filename = "Supply chain.lp")
myModel.optimize()
allVars = myModel.getVars()

print("Optimal Objective: \n" + str(myModel.ObjVal))
print("Optimal Solution:")
allVars = myModel.getVars()
for curVar in allVars:
    print(curVar.varName + " " + str(curVar.x))
```

```
Optimize a model with 26 rows, 160 columns and 320 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e-02, 1e+00]
  Bounds range     [1e+01, 1e+01]
  RHS range        [1e+01, 6e+02]
Presolve removed 10 rows and 12 columns
Presolve time: 0.01s
Presolved: 16 rows, 148 columns, 296 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.1288000e+02   1.315000e+02   0.000000e+00      0s
      15    1.8127000e+02   0.000000e+00   0.000000e+00      0s

Solved in 15 iterations and 0.02 seconds
Optimal objective  1.812700000e+02
Optimal Objective:
181.27
Optimal Solution:
X01 20.0
X02 12.0
X03 30.0
X04 34.0
X05 42.0
X06 88.0
X07 47.0
X08 85.0
X09 50.0
X010 5.0
X11 0.0
X12 10.0
```

```
X13 0.0
X14 0.0
X15 0.0
X16 0.0
X17 0.0
X18 0.0
X19 0.0
X110 10.0
X111 0.0
X112 0.0
X113 0.0
X114 0.0
X115 0.0
X21 0.0
X22 0.0
X23 0.0
X24 0.0
X25 0.0
X26 0.0
X27 2.0
X28 0.0
X29 10.0
X210 0.0
X211 0.0
X212 0.0
X213 0.0
X214 0.0
X215 0.0
X31 0.0
X32 0.0
X33 0.0
X34 0.0
X35 4.0
X36 0.0
X37 0.0
X38 9.0
X39 10.0
X310 0.0
X311 0.0
X312 0.0
X313 0.0
X314 0.0
X315 7.0
X41 0.0
X42 0.0
X43 0.0
X44 10.0
X45 0.0
```

```
X46 0.0
X47 0.0
X48 0.0
X49 0.0
X410 10.0
X411 0.0
X412 4.0
X413 10.0
X414 0.0
X415 0.0
X51 0.0
X52 1.0
X53 0.0
X54 10.0
X55 0.0
X56 0.0
X57 0.0
X58 10.0
X59 0.0
X510 10.0
X511 0.0
X512 0.0
X513 10.0
X514 1.0
X515 0.0
X61 10.0
X62 10.0
X63 10.0
X64 0.0
X65 0.0
X66 0.0
X67 10.0
X68 10.0
X69 10.0
X610 0.0
X611 10.0
X612 10.0
X613 8.0
X614 0.0
X615 0.0
X71 0.0
X72 0.0
X73 0.0
X74 10.0
X75 0.0
X76 2.0
X77 10.0
X78 0.0
```

```
X79 0.0
X710 0.0
X711 5.0
X712 0.0
X713 10.0
X714 10.0
X715 0.0
X81 8.0
X82 0.0
X83 3.0
X84 4.0
X85 10.0
X86 0.0
X87 10.0
X88 0.0
X89 0.0
X810 10.0
X811 0.0
X812 10.0
X813 10.0
X814 10.0
X815 10.0
X91 0.0
X92 0.0
X93 0.0
X94 10.0
X95 10.0
X96 10.0
X97 10.0
X98 0.0
X99 0.0
X910 0.0
X911 0.0
X912 0.0
X913 0.0
X914 0.0
X915 10.0
X101 0.0
X102 0.0
X103 0.0
X104 0.0
X105 0.0
X106 0.0
X107 0.0
X108 0.0
X109 0.0
X1010 5.0
X1011 0.0
```

```
X1012 0.0
X1013 0.0
X1014 0.0
X1015 0.0
```

[ ]:

[ ]:

[ ]:

```
\ Model Supply_chain
\ LP format - for model browsing. Use MPS format to capture full model detail.
Minimize
  0.52 X01 + 0.55 X02 + 0.4 X03 + 0.39 X04 + 0.2 X05 + 0.13 X06 + 0.32 X07
   + 0.14 X08 + 0.12 X09 + 0.69 X010 + 0.29 X11 + 0.04 X12 + 0.62 X13
   + 0.59 X14 + 0.71 X15 + 0.96 X16 + 0.77 X17 + 0.96 X18 + 0.73 X19
   + 0.06 X110 + 0.91 X111 + 0.44 X112 + 0.82 X113 + 0.55 X114 + 0.51 X115
   + 0.19 X21 + 0.31 X22 + 0.99 X23 + 0.53 X24 + 0.74 X25 + 0.12 X26
   + 0.08 X27 + 0.54 X28 + 0.2 X29 + 0.83 X210 + 0.62 X211 + 0.1 X212
   + 0.82 X213 + 0.64 X214 + 0.42 X215 + 0.77 X31 + 0.91 X32 + X33
   + 0.29 X34 + 0.29 X35 + 0.38 X36 + 0.95 X37 + 0.25 X38 + 0.25 X39
   + 0.5 X310 + 0.19 X311 + 0.27 X312 + 0.36 X313 + 0.94 X314 + 0.02 X315
   + 0.03 X41 + 0.64 X42 + 0.48 X43 + 0.23 X44 + 0.68 X45 + 0.76 X46
   + 0.6 X47 + X48 + 0.92 X49 + 0.1 X410 + 0.58 X411 + 0.21 X412
   + 0.13 X413 + 0.98 X414 + 0.23 X415 + 0.51 X51 + 0.56 X52 + 0.92 X53
   + 0.04 X54 + 0.63 X55 + 0.83 X56 + 0.49 X57 + 0.02 X58 + 0.85 X59
   + 0.04 X510 + 0.69 X511 + 0.87 X512 + 0.33 X513 + 0.56 X514 + 0.82 X515
   + 0.15 X61 + 0.47 X62 + 0.14 X63 + 0.85 X64 + 0.82 X65 + 0.78 X66
   + 0.29 X67 + 0.27 X68 + 0.49 X69 + 0.82 X610 + 0.05 X611 + 0.13 X612
   + 0.5 X613 + 0.72 X614 + 0.32 X615 + 0.79 X71 + 0.56 X72 + 0.61 X73
   + 0.03 X74 + 0.53 X75 + 0.13 X76 + 0.23 X77 + 0.9 X78 + 0.98 X79
   + 0.7 X710 + 0.03 X711 + 0.89 X712 + 0.31 X713 + 0.25 X714 + 0.42 X715
   + 0.2 X81 + 0.7 X82 + 0.38 X83 + 0.54 X84 + 0.35 X85 + 0.92 X86
   + 0.47 X87 + 0.77 X88 + 0.83 X89 + 0.53 X810 + 0.62 X811 + 0.19 X812
   + 0.02 X813 + 0.41 X814 + 0.12 X815 + 0.58 X91 + 0.75 X92 + 0.57 X93
   + 0.14 X94 + 0.1 X95 + 0.13 X96 + 0.01 X97 + 0.66 X98 + 0.84 X99
   + 0.75 X910 + 0.39 X911 + 0.6 X912 + 0.55 X913 + 0.99 X914 + 0.04 X915
   + 0.57 X101 + 0.76 X102 + 0.04 X103 + 0.39 X104 + 0.18 X105 + 0.25 X106
   + 0.52 X107 + 0.31 X108 + 0.17 X109 + 0.04 X1010 + 0.81 X1011
   + 0.86 X1012 + 0.58 X1013 + 0.99 X1014 + 0.65 X1015
Subject To
 R0: X11 + X21 + X31 + X41 + X51 + X61 + X71 + X81 + X91 + X101 = 18
 R1: X12 + X22 + X32 + X42 + X52 + X62 + X72 + X82 + X92 + X102 = 21
 R2: X13 + X23 + X33 + X43 + X53 + X63 + X73 + X83 + X93 + X103 = 13
 R3: X14 + X24 + X34 + X44 + X54 + X64 + X74 + X84 + X94 + X104 = 44
 R4: X15 + X25 + X35 + X45 + X55 + X65 + X75 + X85 + X95 + X105 = 24
 R5: X16 + X26 + X36 + X46 + X56 + X66 + X76 + X86 + X96 + X106 = 12
 R6: X17 + X27 + X37 + X47 + X57 + X67 + X77 + X87 + X97 + X107 = 42
 R7: X18 + X28 + X38 + X48 + X58 + X68 + X78 + X88 + X98 + X108 = 29
 R8: X19 + X29 + X39 + X49 + X59 + X69 + X79 + X89 + X99 + X109 = 30
 R9: X110 + X210 + X310 + X410 + X510 + X610 + X710 + X810 + X910 + X1010
   = 45
 R10: X111 + X211 + X311 + X411 + X511 + X611 + X711 + X811 + X911 + X1011
   = 15
 R11: X112 + X212 + X312 + X412 + X512 + X612 + X712 + X812 + X912 + X1012
   = 24
```

= 15
R11: X112 + X212 + X312 + X412 + X512 + X612 + X712 + X812 + X912 + X1012
= 24
R12: X113 + X213 + X313 + X413 + X513 + X613 + X713 + X813 + X913 + X1013
= 48
R13: X114 + X214 + X314 + X414 + X514 + X614 + X714 + X814 + X914 + X1014
= 21
R14: X115 + X215 + X315 + X415 + X515 + X615 + X715 + X815 + X915 + X1015
= 27
R15: – X01 + X11 + X12 + X13 + X14 + X15 + X16 + X17 + X18 + X19 + X110
+ X111 + X112 + X113 + X114 + X115 = 0
R16: – X02 + X21 + X22 + X23 + X24 + X25 + X26 + X27 + X28 + X29 + X210
+ X211 + X212 + X213 + X214 + X215 = 0
R17: – X03 + X31 + X32 + X33 + X34 + X35 + X36 + X37 + X38 + X39 + X310
+ X311 + X312 + X313 + X314 + X315 = 0
R18: – X04 + X41 + X42 + X43 + X44 + X45 + X46 + X47 + X48 + X49 + X410
+ X411 + X412 + X413 + X414 + X415 = 0
R19: – X05 + X51 + X52 + X53 + X54 + X55 + X56 + X57 + X58 + X59 + X510
+ X511 + X512 + X513 + X514 + X515 = 0
R20: – X06 + X61 + X62 + X63 + X64 + X65 + X66 + X67 + X68 + X69 + X610
+ X611 + X612 + X613 + X614 + X615 = 0
R21: – X07 + X71 + X72 + X73 + X74 + X75 + X76 + X77 + X78 + X79 + X710
+ X711 + X712 + X713 + X714 + X715 = 0
R22: – X08 + X81 + X82 + X83 + X84 + X85 + X86 + X87 + X88 + X89 + X810
+ X811 + X812 + X813 + X814 + X815 = 0
R23: – X09 + X91 + X92 + X93 + X94 + X95 + X96 + X97 + X98 + X99 + X910
+ X911 + X912 + X913 + X914 + X915 = 0
R24: – X010 + X101 + X102 + X103 + X104 + X105 + X106 + X107 + X108 + X109
+ X1010 + X1011 + X1012 + X1013 + X1014 + X1015 = 0
R25: X01 + X02 + X03 + X04 + X05 + X06 + X07 + X08 + X09 + X010 <= 615
Bounds
X11 <= 10
X12 <= 10
X13 <= 10
X14 <= 10
X15 <= 10
X16 <= 10
X17 <= 10
X18 <= 10
X19 <= 10
X110 <= 10
X111 <= 10
X112 <= 10
X113 <= 10

```
X215 <= 10
X31 <= 10
X32 <= 10
X33 <= 10
X34 <= 10
X35 <= 10
X36 <= 10
X37 <= 10
X38 <= 10
X39 <= 10
X310 <= 10
X311 <= 10
X312 <= 10
X313 <= 10
X314 <= 10
X315 <= 10
X41 <= 10
X42 <= 10
X43 <= 10
X44 <= 10
X45 <= 10
X46 <= 10
X47 <= 10
X48 <= 10
X49 <= 10
X410 <= 10
X411 <= 10
X412 <= 10
X413 <= 10
X414 <= 10
X415 <= 10
X51 <= 10
X52 <= 10
X53 <= 10
X54 <= 10
X55 <= 10
X56 <= 10
X57 <= 10
X58 <= 10
X59 <= 10
X510 <= 10
X511 <= 10
X512 <= 10
X513 <= 10
X514 <= 10
X515 <= 10
```

X512 <= 10
X513 <= 10
X514 <= 10
X515 <= 10
X61 <= 10
X62 <= 10
X63 <= 10
X64 <= 10
X65 <= 10
X66 <= 10
X67 <= 10
X68 <= 10
X69 <= 10
X610 <= 10
X611 <= 10
X612 <= 10
X613 <= 10
X614 <= 10
X615 <= 10
X71 <= 10
X72 <= 10
X73 <= 10
X74 <= 10
X75 <= 10
X76 <= 10
X77 <= 10
X78 <= 10
X79 <= 10
X710 <= 10
X711 <= 10
X712 <= 10
X713 <= 10
X714 <= 10
X715 <= 10
X81 <= 10
X82 <= 10
X83 <= 10
X84 <= 10
X85 <= 10
X86 <= 10
X87 <= 10
X88 <= 10
X89 <= 10
X810 <= 10
X811 <= 10
X812 <= 10

```
X82 <= 10
X83 <= 10
X84 <= 10
X85 <= 10
X86 <= 10
X87 <= 10
X88 <= 10
X89 <= 10
X810 <= 10
X811 <= 10
X812 <= 10
X813 <= 10
X814 <= 10
X815 <= 10
X91 <= 10
X92 <= 10
X93 <= 10
X94 <= 10
X95 <= 10
X96 <= 10
X97 <= 10
X98 <= 10
X99 <= 10
 X910 <= 10
 X911 <= 10
 X912 <= 10
 X913 <= 10
 X914 <= 10
 X915 <= 10
 X101 <= 10
 X102 <= 10
 X103 <= 10
 X104 <= 10
 X105 <= 10
 X106 <= 10
 X107 <= 10
 X108 <= 10
 X109 <= 10
 X1010 <= 10
 X1011 <= 10
 X1012 <= 10
 X1013 <= 10
 X1014 <= 10
 X1015 <= 10
End

```